Algorithm 1 Preprocessing of benchmark dataset

Input: A set of audio files $\mathcal{F}_\mu = \{f_1, f_2, \dots, f_n\}$ with labels $\mathcal{L} = \{\mathrm{HC}, \mathrm{PD}\}$; segment duration T_s (in seconds); sampling rate f_s (Hz); noise duration T_{noise}; spectral threshold coefficient α
Output: A set of segmented audio signals $\mathcal{Y}_{\mathrm{segments}}$ for each processed file
Step 1: Data Preprocessing (Noise Reduction)
for each audio file $f \in \mathcal{F}_\mu$ **do**
  Load the audio file: $(y(t), f_s) \leftarrow \mathrm{load}(f)$
  Sample the speech signal at f_s to obtain discrete samples $y[n]$
  Spectral Gating:
  Compute the Short-Time Fourier Transform (STFT):

$$Y[k,m] = \mathrm{STFT}(y[n]) = \sum_n y[n] \cdot w[n-m] e^{-j2\pi kn/N}$$

  Estimate the noise spectrum from the initial segment $y_{\mathrm{noise}}[n]$ (duration T_{noise}):

$$N[k] = \frac{1}{M} \sum_{m=1}^{M} |Y[k,m]|$$

  Apply spectral threshold $T[k] = \alpha N[k]$, where α is the threshold ratio
  for each frequency bin k and time frame m **do**
    Perform spectral gating:

$$Y_{\mathrm{denoised}}[k,m] = \begin{cases} Y[k,m], & \text{if } |Y[k,m]| \geq T[k] \\ 0, & \text{otherwise} \end{cases}$$

  end for
  Compute the Inverse STFT to obtain the denoised signal:

$$y_{\mathrm{denoised}}[n] = \mathrm{ISTFT}(Y_{\mathrm{denoised}}[k,m])$$

end for

Step 2: Audio Segmentation
for each denoised signal $y^i_{\mathrm{denoised}}[n] \in \mathcal{Y}_{\mathrm{denoised}}$ **do**
  Compute segment length $S = T_s \times f_s$
  Divide the signal into $L = \left\lfloor \frac{N}{S} \right\rfloor$ segments
  for each segment $j = 1$ to L **do**
    Compute start and end indices:

$$\mathrm{start}_j = (j-1) \times S, \quad \mathrm{end}_j = j \times S$$

    Extract the segment:

$$y_j[n] = y^i_{\mathrm{denoised}}[n], \quad n \in [\mathrm{start}_j, \mathrm{end}_j)$$

    Append $y_j[n]$ to $\mathcal{Y}_{\mathrm{segments}}$
  end for
end for
return Segmented audio signals $\mathcal{Y}_{\mathrm{segments}}$

Algorithm 2 Tabular Feature Extraction

Input: Denoised audio signals $\mathcal{Y}_{\text{segments}} = \{y^1_{\text{denoised}}, y^2_{\text{denoised}}, \dots, y^n_{\text{denoised}}\}$; segment duration T_s; sampling rate f_s
Output: Feature matrix X with extracted features for each segment
Step 2: Compute Features for Each Segment
for each segment $y_j[n] \in \mathcal{Y}_{\text{segments}}$ **do**
  Statistical Features:
  Compute energy:

$$E_j = \sqrt{\frac{1}{S}\sum_{n=1}^{S} y_j[n]^2}$$

  Compute entropy of energy:

$$H_j = -\sum_{n=1}^{S} p_n \log(p_n), \quad p_n = \frac{y_j[n]^2}{\sum_{n=1}^{S} y_j[n]^2}$$

  Compute kurtosis and skewness:

$$\text{Kurtosis}_j = \frac{\frac{1}{S}\sum_{n=1}^{S}(y_j[n]-\mu_j)^4}{\sigma_j^4} - 3$$

$$\text{Skewness}_j = \frac{\frac{1}{S}\sum_{n=1}^{S}(y_j[n]-\mu_j)^3}{\sigma_j^3}$$

  Praat Features:
  Compute jitter, shimmer, fundamental frequency (F_0), and harmonic-to-noise ratio (HNR):

$$J_{\text{local},j}, S_{\text{local},j}, F_{0,j}, \text{HNR}_j$$

  Spectral Features:
  Compute zero-crossing rate:

$$\text{ZCR}_j = \frac{1}{S-1}\sum_{n=1}^{S-1}\left|\text{sgn}(y_j[n]) - \text{sgn}(y_j[n+1])\right|$$

  Compute spectral centroid:

$$SC_j = \frac{\sum_k k \cdot |Y_j[k]|}{\sum_k |Y_j[k]|}$$

  Compute Mel-Frequency Cepstral Coefficients (MFCCs):

$$\text{MFCCs}_j = \text{DCT}\left(\log\left(\text{MelSpectrum}(y_j[n])\right)\right)$$

  Append features $\left[E_j, H_j, \text{Kurtosis}_j, \text{Skewness}_j, J_{\text{local},j}, S_{\text{local},j}, F_{0,j}, \text{HNR}_j, \text{ZCR}_j, SC_j, \text{MFCCs}_j\right]$ to feature matrix X
end for
return Feature matrix X

Algorithm 3 Speech Signal to Spectrogram

Input: Audio files $F_\mu = \{f_1, f_2, \dots, f_n\}$, segment duration $T_s = 5$ seconds
Output: Spectrogram images (STFT, Mel, MFCC, CQT, Chroma)
Step 1: Audio Segmentation
for each file $f \in F_\mu$ **do**
    Load audio file $y, sr = \text{load}(f)$
    Compute segment length $S = T_s \times sr$
    for each chunk i from 1 to $\left\lfloor \frac{N}{S} \right\rfloor$ **do**
        Extract chunk $y_i = y[\text{start}_i : \text{end}_i]$
    end for
end for
Step 2: Noise Reduction Using Spectral Gating
for each chunk y_i **do**
    Extract noise profile $y_{\text{noise}} = y[0 : N_p]$
    Compute STFT of noise profile M_{noise}
    Apply spectral threshold to filter M_{denoised}
end for
Step 3: Spectrogram Generation
for each denoised chunk y_{denoised} **do**
    Compute STFT, Mel-spectrogram, MFCC, CQT, Chroma
end for
Step 4: Save Spectrograms
for each spectrogram type S **do**
    Save corresponding image to output directory
end for